

# Apache Spark - Practice set 1

Total points **15/60**

The respondent's email (baburam@fusemachines.com) was recorded on submission of this form.

Question 1: Determine if the following statement is true or false. When using `DataFrame.persist()` data on disk is always serialized. \*

1/1

FALSE

TRUE

Correct

Question 2: If we want to create a constant integer 1 as a new column 'new\_column' in a dataframe df, which code block we should select ? \*

1/1

`df.withColumn("new_column", lit(1))`

Correct

`df.withColumn(new_column, lit(1))`

`df.withColumn("new_column", lit("1"))`

`df.withColumn("new_column", 1)`

`df.withColumnRenamed('new_column', lit(1))`

Question 3: Choose the equivalent code block to: `df.filter(col("count") < 2)` Where df is a valid dataframe which has a column named count \*

0/1

`df.where("count is smaller then 2").show(2)`

Incorrect

`df.getWhere("count < 2")`

`df.where(count < 2)`

`df.where("count < 2")`

`df.select("count < 2")`

Correct answer

`df.where("count < 2")`

Question 4: What is the correct syntax to run sql queries programmatically ? \*

0/1

spark.query()  
Incorrect

It is not possible to run sql queries programmatically

spark.sql()  
spark.run()  
spark.runSql()  
Correct answer  
spark.sql()

Question 5: What command we can use to get the number of partition of a dataframe named df ? \*

1/1  
df.rdd.getNumPartitions()  
Correct

df.getPartitionSize()  
df.rdd.getPartitionSize()  
df.getNumPartitions()

Question 6: Select the code block which counts the number of “quantity” for each “invoiceNo” in the dataframe df. \*

0/1  
df.groupBy(InvoiceNo).agg( expr("count(Quantity)"))  
Incorrect

df.groupBy("InvoiceNo").agg( expr("count(Quantity)"))  
df.groupBy(InvoiceNo).agg( expr(count(Quantity)))  
df.reduceBy("InvoiceNo").agg( expr("count(Quantity)"))  
df.groupBy("InvoiceNo").agg( expr(count(Quantity)))  
Correct answer  
df.groupBy("InvoiceNo").agg( expr("count(Quantity)"))

Question 7: Which of the following describe optimizations enabled by adaptive query execution (AQE)?  
Choose two. \*

0/1  
AQE allows you to dynamically switch join strategies.  
Correct

AQE allows you to dynamically convert physical plans to RDDs

AQE allows you to dynamically select physical plans based on cost.

AQE allows you to dynamically coalesce shuffle partitions

AQE allows you to dynamically reorganize query orders.

**Required**

**Correct answer**

AQE allows you to dynamically switch join strategies.

AQE allows you to dynamically coalesce shuffle partitions

**Question 8:** The code block shown below intends to return a new DataFrame with column “old” renamed to “new” but it contains an error. Identify the error. `df.withColumnRenamed(“new”, “old”) *`  
0/1

We need to add ‘col’ to specify that it’s a column. `df.withColumnRenamed(col(“new”), col(“old”))`

**Incorrect**

There should be no quotes for the column names. `df.withColumnRenamed(new, old)`

`WithColumnRenamed` is not a valid function, we need to use `df.withColumnRenamed(“new”, “old”)`

Parameters are inverted; correct usage is `df.withColumnRenamed(“old”, “new”)`

**Correct answer**

Parameters are inverted; correct usage is `df.withColumnRenamed(“old”, “new”)`

**Question 9:** Given an instance of `SparkSession` named `spark`, and the following DataFrame named \*  
0/1



Captionless Image

Option 1

**Incorrect**



Option 2



Option 3



Option 4

**Correct answer**

Option 4

**Question 10:** Choose invalid execution mode in the following responses. \*  
0/1

Local

**Incorrect**

Client

Standalone

Cluster

Correct answer

Standalone

Question 11: You have a need to transform a column named 'date' to a timestamp format. Assume that the column 'date' is timestamp compatible. You have written the code block down below, but it contains an error. Identify and fix it. `df.select(to_timestamp(col("date")).show() *`

0/1

We need to add a format and it should be the first parameter passed to this function. `df.select(to_timestamp('yyyy-dd-MM', col("date")))`

Incorrect

`to_timestamp()` is not a valid operation. Proper function is `toTimestamp()` and also we need to add a format.

`df.select(toTimestamp(col("date"), 'yyyy-dd-MM'))`

`to_timestamp()` is not a valid operation. Proper function is `toTimestamp()` `df.select(toTimestamp(col("date")))`

`to_timestamp` requires always a format ! So you need to add one `df.select(to_timestamp(col("date"), 'yyyy-dd-MM'))`

Correct answer

`to_timestamp` requires always a format ! So you need to add one `df.select(to_timestamp(col("date"), 'yyyy-dd-MM'))`

Question 12: Spark dynamically handles skew in sort-merge join by splitting (and replicating if needed) skewed partitions. Which property need to be enabled to achieve this ? \*

0/1

`spark.sql.adaptive.optimize.skewJoin`

Incorrect

`spark.sql.adaptive.skewJoin.enable`

`spark.sql.skewJoin.enabled`

`spark.sql.adaptive.skewJoin.enabled`

Correct answer

`spark.sql.adaptive.skewJoin.enabled`

Question 13: The code block shown below should return a new DataFrame with 25 percent of random records from dataframe `df` without replacement. Choose the response that correctly fills in the numbered blanks within the code block to complete this task. Code block: `df._1_( _2_, _3_, _4_ ) *`

0/1



Option 1

Incorrect



Option 2



Option 3



Option 4



Option 5



Option 6

**Correct answer**

Option 5

Question 14: Which of the following code blocks changes the parquet file content given that there is already a file exist with the name that we want to write ? \*

1/1

```
df.write.mode("overwrite").option("compression", "snappy").save("path")
```

**Correct**

```
df.write.format("parquet").option("compression", "snappy").path("path")
```

```
df.save.format("parquet").mode("overwrite").option("compression", "snappy").path("path")
```

Question 15: Which of the following DataFrame operation is classified as a narrow transformation ? \*

0/1

```
repartition()
```

**Incorrect**

```
distinct()
```

```
orderBy()
```

```
filter()
```

```
coalesce()
```

**Correct answer**

```
filter()
```

Question 16: tableA is a DataFrame consisting of 20 fields and 40 billion rows of data with a surrogate key field. tableB is a DataFrame functioning as a lookup table for the surrogate key consisting of 2 fields and 5,000 rows. If the in-memory size of tableB is 22MB, what occurs when the following code is executed: ? `df = tableA.join(tableB, "primary_key")` \*

0/1

The contents of tableB will be replicated and sent to each executor to eliminate the need for a shuffle stage during the join.

**Incorrect**

A non-broadcast join will be executed with a shuffle phase since the broadcast table is greater than the 10MB default threshold and the broadcast hint was not specified.

An exception will be thrown due to tableB being greater than the 10MB default threshold for a broadcast join.

The contents of tableB will be partitioned so that each of the keys that need to be joined on in tableA partitions on each executor will match.

**Correct answer**

A non-broadcast join will be executed with a shuffle phase since the broadcast table is greater than the 10MB default threshold and the broadcast hint was not specified.

**Question 17: What won't cause a full shuffle knowing that dataframe 'df' has 8 partitions ? \***

0/1

`df.repartition(12)`

Incorrect

`df.coalesce(4)`

All of them will cause a full shuffle.

**Correct answer**

`df.coalesce(4)`

**Question 18: Which of the following transformation is not evaluated lazily ? \***

0/1

`sample()`

Incorrect

None of the responses, all transformations are lazily evaluated.

`repartition()`

`filter()`

`select()`

**Correct answer**

None of the responses, all transformations are lazily evaluated.

**Question 19: Which of the following is true for driver ? \***

0/1

Responsible for executing work that will be completed in parallel.

Incorrect

Is a chunk of data that sit on a single machine in a cluster.

Responsible for assigning work that will be completed in parallel.

Responsible for allocating resources for worker nodes.

Reports the state of some computation back to a central system.

**Correct answer**

Responsible for assigning work that will be completed in parallel.

**Question 20:** The code block shown below should return a DataFrame with column only aSquared dropped from DataFrame df. Choose the response that correctly fills in the numbered blanks within the code block to complete this task.

Code block: df.\_\_1\_\_(\_\_2\_\_) \*

1/1



Option 1

Correct



Option 2



Option 3



Option 4

**Question 21:** The goal of Dynamic Partition Pruning (DPP) is to allow you to read only as much data as you need. Which property needs to be set in order to use this functionality ? \*

0/1

spark.sql.dynamicPartitionPruning.optimizer.enabled

Incorrect

spark.sql.optimizer.dynamicPartitionPruning.enabled

spark.sql.adaptive.dynamicPartitionPruning.enabled

spark.sql.dynamicPartitionPruning.enabled

**Correct answer**

spark.sql.optimizer.dynamicPartitionPruning.enabled

**Question 22:** The code block shown below contains an error. The code block is intended to write a text file in the path. Identify the error. \*

1/1



Captionless Image

For text files, we can only have one column in the dataframe that we want to write.

Correct

The maximum limit of lines in a text file has been reached and therefore we cannot create a text file.

We need to use save instead of write function.

We need to provide at least one option.

Question 23: If spark is running in cluster mode, which of the following statements about nodes is incorrect ? \*

1/1

The spark driver runs in its own non-worker node without any executors

Correct

There is one single worker node that contains the Spark driver and the executors

There might be more executors than total number of nodes

Each executor is running in a JVM inside of a worker node

There is at least one worker node in the cluster

Question 24: Which property is used to scale up and down dynamically based on applications' current number of pending tasks in a spark cluster ? \*

0/1

There is no need to set a property since spark is by default capable of resizing

Incorrect

Dynamic allocation

Fair Scheduler

Correct answer

Dynamic allocation

Question 25: Which of the following describes the relationship between worker nodes and executors? \*

1/1

An executor is a Java Virtual Machine (JVM) running on a worker node.

Correct

Executors and worker nodes are not related.

There are always the same number of executors and worker nodes.

There are always more worker nodes than executors.

A worker node is a Java Virtual Machine (JVM) running on an executor.

Question 26: The code blown down below intends to join df1 with df2 with inner join but it contains an error. Identify the error.

df1.join(df2, "inner", df1.col("id") == df2.col("id")) \*



1/1

The join type is not in right order. The correct query should be `d1.join(d2, d1.col("id") == df2.col("id"), "inner")`

Correct

There should be two `==` instead of `===`. So the correct query is `d1.join(d2, "inner", d1.col("id") == df2.col("id"))`

We cannot do inner join in spark 3.0, but it is in the roadmap.

`d1.join(d2, d1.col("id") === df2.col("id"), "inner")`

Question 27: What are the possible strategies in order to decrease garbage collection time ? \*

0/1

Create fewer objects

Correct

Persist objects in serialized form

Increase java heap space size

Required

Correct answer

Create fewer objects

Persist objects in serialized form

Increase java heap space size

Question 28: There is a global temp view named 'my\_global\_view'. If I want to query this view within spark, which command I should choose ? \*

0/1

`spark.read.table("my_global_view")`

Incorrect

`spark.read.view("my_global_view")`

`spark.read.table("global_temp.my_global_view")`

`spark.read.view("global_temp.my_global_view")`

Correct answer

`spark.read.table("global_temp.my_global_view")`

Question 29: The code block shown below contains an error. Identify the error. \*

1/1

Captionless Image



Option 1

Correct

There is no column id created in the database.



Option 3



Option 4

There is no error in the code.

Question 30: Which of the following code blocks reads from a csv file where values are separated with ‘;’ ? \*

0/1

`spark.read.format("csv").option("header", "true").option("inferSchema", "true").load(file)`

Incorrect

`spark.read.format("csv").option("header", "true").option("inferSchema", "true").option("sep", ";").load(file)`

`spark.read.format("csv").option("header", "true").option("inferSchema", "true").option("sep", "true").toDf(file)`

`spark.load.format("csv").option("header", "true").option("inferSchema", "true").read(file)`

Correct answer

`spark.read.format("csv").option("header", "true").option("inferSchema", "true").option("sep", ";").load(file)`

Question 31: Which of the following 3 DataFrame operations are NOT classified as an action? Choose 3 answers: \*

0/1

`show()`

Incorrect

`foreach()`

`first()`

`printSchema()`

`limit()`

`cache()`

Required

Correct answer

`printSchema()`

`limit()`

`cache()`

Question 32: At which stage do the first set of optimizations take place? \*

0/1


Physical Planning

Incorrect

Logical Optimization  
Analysis  
Code Generation  
Correct answer  
Logical Optimization

Question 33: Which of the following statements regarding caching are true? \*

0/1

Captionless Image

Red, Blue, and White

Incorrect

Red, White and Green

Green and White

Green and Blue

Correct answer

Red, White and Green

Question 34: Which of the following operations can be used to create a new DataFrame with a new column and all previously existing columns from an existing DataFrame ? \*

0/1

DataFrame.filter()

Incorrect

DataFrame.withColumnRenamed()

DataFrame.drop()

DataFrame.withColumn()

DdataFrame.head()

Correct answer

DataFrame.withColumn()

Question 35: Which of the following 3 DataFrame operations are classified as a wide transformation ?  
Choose 3 answers: \*

0/1

drop()

Incorrect

orderBy()

distinct()

```
repartition()
```

```
filter()
```

```
cache()
```

**Required**

**Correct answer**

```
orderBy()
```

```
distinct()
```

```
repartition()
```

Question 36: When joining two dataframes, if there is a need to evaluate the keys in both of the DataFrames or tables and include all rows from the left DataFrame as well as any rows in the right DataFrame that have a match in the left DataFrame also If there is no equivalent row in the right DataFrame, we want to insert null: which join type we should select ? `df1.join(person, joinExpression, joinType) *`

1/1

```
joinType = "left_outer"
```

**Correct**

```
joinType = "left_semi"
```

```
joinType = "leftAnti"
```

```
joinType = "leftOuter"
```

Question 37: You have a need to sort a dataframe named `df` which has some null values on column `a`. You want the null values to appear first, and then the rest of the rows should be ordered descending based on the column `a`. Choose the right code block to achieve your goal. \*

0/1

```
df.orderBy(desc("a"))
```

**Incorrect**

```
df.orderBy(desc_nulls_first(a))
```

It is not possible to sort, when there are null values on the specified column.

```
df.sortBy(desc_nulls_first("a"))
```

```
df.orderBy(df.a.desc_nulls_first())
```

**Correct answer**

```
df.orderBy(df.a.desc_nulls_first())
```

Question 38: Which of the following code blocks concatenates two DataFrames `df1` and `df2` ? \*

0/1

```
df1.addAll(df2)
```

**Incorrect**

```
df1.union(df2)
df1.append(df2)
df1.appendAll(df2)
df1.add(df2)
Correct answer
df1.union(df2)
```

Question 39: Which of the following code blocks returns a DataFrame with a new column aSquared and all previously existing columns from DataFrame df given that df has a column named a ? \*

0/1

```
df.withColumn(aSquared, col("a") * col("a"))
```

Incorrect

```
df.withColumn("aSquared", col("a") * col("a"))
df.withColumn(col("a") * col("a"), "aSquared")
df.withColumn("aSquared", col(a) * col(a))
df.withColumn(aSquared, col(a) * col(a))
```

Correct answer

```
df.withColumn("aSquared", col("a") * col("a"))
```

Question 40: Which of the following statement is true for broadcast variables ? \*

0/1

The canonical use case is to pass around a extremely large table that does not fit in memory on the executors.

Incorrect

It provides a mutable variable that a Spark cluster can safely update on a per-row basis

It is a way of updating a value inside of a variety of transformations and propagating that value to the driver node in an efficient and fault-tolerant way.

Broadcast variables are shared, immutable variables that are cached on every machine in the cluster instead of serialized with every single task

Correct answer

Broadcast variables are shared, immutable variables that are cached on every machine in the cluster instead of serialized with every single task

Question 41: Which of the following describes a worker node ? \*

0/1

Worker nodes always have a one-to-one relationship with executors.

Incorrect

Worker nodes are synonymous with executors.

Worker nodes are the most granular level of execution in the Spark execution hierarchy.

Worker nodes are the nodes of a cluster that perform computations.

**Correct answer**

Worker nodes are the nodes of a cluster that perform computations.

**Question 42:**How make sure that dataframe df has 12 partitions given that df has 4 partitions ? \*

0/1

df.setPartition(12)

**Incorrect**

df.setPartition()

df.repartition(12)


df.repartition()

**Correct answer**

df.repartition(12)

**Question 43:**Given an instance of SparkSession named spark, reviewing the following code what's the output ? \*

0/1

Captionless Image



Option 1

**Incorrect**



Option 2



Option 3



Option 4

**Correct answer**

Option 2

**Question 44:**Which of the followings are useful use cases of spark ? \*

0/1

Analyzing graph data sets and social networks

**Incorrect**

Building, training, and evaluating machine learning models using MLlib

Processing in parallel large data sets distributed across a cluster

Performing ad hoc or interactive queries to explore and visualize data sets


All of the answers are correct.

Correct answer

All of the answers are correct.

Question 45: Choose the right order of commands in order to query table 'test' in database 'db' \*

0/1

Captionless Image

3, 4

Incorrect

2, 4

1, 5

1, 4

2, 5

3, 5

Correct answer

1, 4

Question 46: Your application on production is crashing lately and your application gets stuck at the same level every time you restart the spark job . You know that it is the toLocalIterator function is causing the problem. What are the possible solutions to this problem ? \*

0/1

Reduce the memory of the driver

Incorrect

Reduce the size of your partitions if possible.

Use collect function instead of toLocalIterator

There is nothing to worry, application crashes are expected and will not affect your application at all.

Correct answer

Reduce the size of your partitions if possible.

Question 47: We want to create a dataframe with a schema. Choose the correct order in order to achieve this goal. \*

1/1

Captionless Image

1, 2, 3

Correct

1, 2, 4

1, 2, 6

1, 2, 5

Question 48: Given that the number of partitions of dataframe df is 4 and we want to write a parquet file in a given path. Choose the correct number of files after a successful write operation. \*

0/1

1

Incorrect

4

8

Correct answer

4

Question 49: If we want to store RDD as deserialized Java objects in the JVM and if the RDD does not fit in memory, store the partitions that don't fit on disk, and read them from there when they're needed also replicate each partition on two cluster nodes, which storage level we need to choose ? \*

1/1

MEMORY\_AND\_DISK\_2

Correct

MEMORY\_AND\_DISK\_2\_SER

MEMORY\_AND\_DISK

MEMORY\_ONLY\_2

Question 50: If spark is running in client mode, which of the following statement about is correct ? \*

0/1

Spark driver is randomly attributed to a machine in the cluster

Incorrect

Spark driver remains on the client machine that submitted the application

Spark driver is attributed to the machine that has the most resources

The entire spark application is run on a single machine.

Correct answer

Spark driver remains on the client machine that submitted the application

Question 51: Which of the following are correct for slots ? \*

0/1

Each slot can be assigned a task.

Correct



It is interchangeable with tasks.

Each executor has a number of slots.

Spark parallelizes via slots.

All of the answers are correct.

**Required**

**Correct answer**


Each slot can be assigned a task.

Each executor has a number of slots.

Spark parallelizes via slots.

**Question 52:** Consider the following DataFrame: \*

1/1

 Captionless Image



Option 1

Correct



Option 2



Option 3



Option 4

**Question 53:** What causes a stage boundary ? \*

0/1

Failure of driver node

Incorrect

Failure of network

Shuffle

Failure of worker node

**Correct answer**

Shuffle

**Question 54:** Let's suppose that we have a dataframe with a column 'today' which has a format 'YYYY-MM-DD'. You want to add a new column to this dataframe 'week\_ago' and you want its value to be one week prior to column 'today'. Select the correct code block. \*

0/1

```
df.withColumn("week_ago", week_sub(col("today"), 7))
```

Incorrect

```
df.withColumn(week_ago, date_sub(col("today"), 7))
df.withColumn( date_sub(col("today"), 7), "week_ago")
df.withColumn("week_ago", col("today")- 7))
df.withColumn("week_ago", date_sub(col("today"), 7))
```

Correct answer

```
df.withColumn("week_ago", date_sub(col("today"), 7))
```

Question 55:Which of the following describes a job best ? \*

0/1

A unit of work that will be sent to one executor.

Incorrect

An external service for acquiring resources on the cluster (e.g. standalone manager, Mesos, YARN)

User program built on Spark. Consists of a driver program and executors on the cluster.

A parallel computation consisting of multiple tasks that gets spawned in response to a Spark action (e.g. save, collect).

A process launched for an application on a worker node, that runs tasks and keeps data in memory or disk storage across them.

Correct answer

A parallel computation consisting of multiple tasks that gets spawned in response to a Spark action (e.g. save, collect).

Question 56:Which of the following statements about Spark accumulator variables is NOT true? \*

0/1

In transformations, each task's update can be applied more than once if tasks or job stages are re-executed.

Incorrect

The Spark UI displays all accumulators used by your application.

For accumulator updates performed inside actions only, Spark guarantees that each task's update to the accumulator will be applied only once, meaning that restarted tasks will not update the value.

Accumulators provide a shared, mutable variable that a Spark cluster can safely update on a per-row basis.

You can define your own custom accumulator class by extending `org.apache.spark.util.AccumulatorV2` in Java or Scala or `pyspark.AccumulatorParam` in Python.

Correct answer

The Spark UI displays all accumulators used by your application.

Question 57:The following statement will create a managed table `dataframe.write.option('path', '/my_paths/').saveAsTable("managed_my_table")` \*

0/1

TRUE

Incorrect

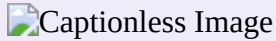
FALSE

Correct answer

FALSE

reviewQuestion 58: Given the code block down below, a database test and a dataframe containing nulls, identify the error. \*

0/1



We need to use function 'query' instead of 'sql' to query table test.

Incorrect

This WHERE clause does not guarantee the strlen UDF to be invoked after filtering out nulls. So we will have null pointer exception.

There is no problem with this query.

Correct answer

This WHERE clause does not guarantee the strlen UDF to be invoked after filtering out nulls. So we will have null pointer exception.

Question 59: For the following dataframe if we want to fully cache the dataframe, what functions should we call in order ?df = spark.range(1 \* 10000000).toDF("id") \*

1/1

df.cache() and then df.count()

Correct

Only df.cache()

df.cache() and then df.take(1)

Only df.count()

df.take(1)

Question 60: The code block shown below should return a new DataFrame with a new column named "casted" who's value is the long equivalent of column "a" which is a integer column also this dataframe should contain all the previously existing columns from DataFrame df. Choose the response that correctly fills in the numbered blanks within the code block to complete this task. Code block:

df.\_1\_(2\_) \*

0/1



Option 1

Incorrect



Option 2



Option 3



Option 4



Option 5



Option 6

**Correct answer**

Option 3

This form was created inside of fusemachines.



[Google Forms](#)